CS2106: Operating Systems
AY19/20 Semester 1
Introduction to OS labs

Section 1. Introduction

If you used Unix or one of the Unix variants before, then most of the introductions below should be familiar to you. Feel free to skip all the parts which you already understood. For example, if you used SoC's unix server ***sunfire*** (by SSH into `sunfire.comp.nus.edu.sg`) before in other modules, then the main difference is just the graphical user interface (GUI). Please note that **Solaris** (a Unix variant by Oracle) is used on the sunfire server while **Linux** is used in the OS Lab.

1.1 Logging In

We have created a number of user accounts in the OS lab server. The username has the form of ***e0123456***, same with your NUSNET id. Once the lab allocation is over, we will upload your account and password into the LumiNUS gradebook as a way to distribute this information securely. Please use **only your allocated account.**

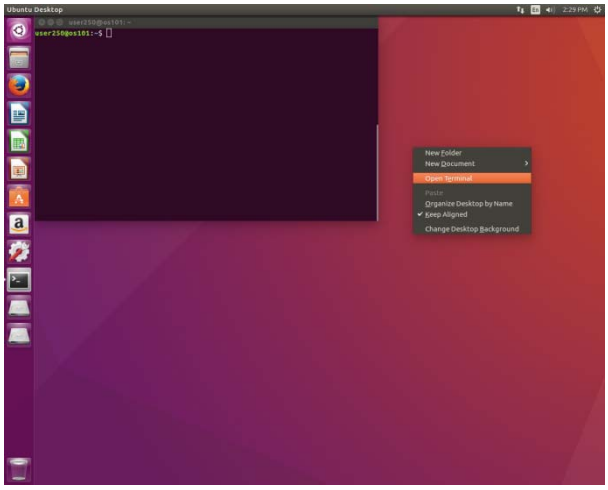
To login to linux, you must give your username followed by your password:

```
login: e0123456
password: <found in Gradebook>
```

The lab PCs use the Ubuntu 16.04 distribution of Linux (http://www.ubuntu.com/download/desktop). A Linux distribution is essentially a software collection with Linux kernel at its core and a number of popular applications.

After entering your password, you will be brought to the desktop GUI. Depending on the setup and whether you have any previously saved sessions, you may see different desktops and windows open. The graphical interface and desktop you see in Linux is very customizable. The basic graphical system is known as ***X Windows*** which provides the underlying services. The actual interface you normally interact with is provided by a ***graphical window manager*** (you can take it as the "desktop" for simplicity). There are a number of graphical window managers that provides very different and customizable user experience on Linux. For example, the **GNOME** and **KDE** are two very popular window managers.

Ubuntu 16.04 comes with ***Unity*** as the default desktop. In Linux, while there is a GUI, it is still common to make use of the text based **c**ommand **l**ine **i**nterface (**CLI**) and programs.



There are two simple ways to get the CLI in Ubuntu:

a. Right click on the desktop and choose "Open Terminal".

b. Press Ctrl-Alt-t to open the terminal directly.

At this point, you should see something similar to the screenshot on the left.

1.2 Changing your Password

You **must** now change the initial password to your own personal password. Choose a password which you can remember. PLEASE DO NOT LOSE THE PASSWORD. Use the **`yppasswd`** command to change your password. An example usage is given below:

```
[userXXX@os101~]$ yppasswd
Changing NIS account information for userXXX on
hostYYY.comp.nus.edu.sg
Please enter old password:

Changing NIS account for userXXX on hostYYY.comp.nus.edu.sg
Please enter new password:
Please retype new password:
The NIS password has been changed on hostYYY.comp.nus.edu.sg
```

1.3 User Interfaces

You can interact with Linux via the **c**ommand **l**ine **i**nterface (CLI) or graphical desktop interface. We will mostly use only the **CLI** in the labs.

1.3.1 The Command Line Interface (CLI)

The terminal you started is actually composed of two programs. One is a graphical application which gives a window and a terminal emulation. The other is a **shell** which interprets ASCII command lines which is connected to the terminal. The **shell** uses a command line interface, because it is text based. You type in commands to a shell which performs the command(s) in that line of text. In Windows, there is also a command line shell, **`cmd`** which has a similar purpose. There are a number of different shells available under Linux. **BASH** (**B**ourne **A**gain **SH**ell) is the default and is a GNU enhancement descended from the original Unix shell, the **Bourne shell** (**sh**). A list of shells of typical shells in Unix are: **`sh, bash, csh, ksh, tcsh, zsh`**. (Not all these shells may be available). You can stick to the default choice of **`bash`** unless you feel like trying out other shell(s). For interactive usage, **`bash`** and **`tcsh`** are the popular shells. Shells are also used for writing shell scripts (this may be a non-interactive use of the shell) which are small programs in the shell language and some of the other shells can be convenient/efficient for that purpose. If you want to change you default shell, you can use the **`chsh`** program.

1.3.2 The X11 GUI

Linux also has a GUI (graphical user interface) which is analogous to the Windows graphical shell. The GUI consists of:

- The X Window System which provides the core graphics and windowing mechanisms.
- A window manager and desktop environment. It is possible to run different window managers which give different GUIs and look and feel on top of X.

You can play around with the desktop and try the various programs.

1.4 Using the built-in manual page (`man`)

Most Unix commands and C built-in libraries have built-in documentation. Use the *Unix Manual* command: **`man`** to access these documentation:

```
man XXXX
```

where **`XXXX`** is either the command/C built-in library call/etc that you want to know more about.
For example, it is self-documenting:

```
$ man man
```

The above means “get the manual page for the command **`man`**”

Try out:

```
$ man fprintf
```

Which explain the C-Library function **fprintf().** Other commands which you may have tried are also in the manual page, eg. **man date**. Take note that Unix man pages are written in a terse and very technical fashion. So it works more like a reference rather than a user-friendly tutorial. For a taste of a very long and scary looking man page, try "**man gcc**"!

The manual pages are organized using sections, and an entry might be in more than one section, e.g. **man 1 login** documents the command line login program while **man 3 login** documents the files which record the users of the system. You can search for keywords with the **-k** option, eg. **man -k login**.

The various manual sections have their own introduction. Try **man intro** or **man 1 intro**. System calls are described in **man 2 intro**. The **man** uses a **pager**, which is another program to display one page at a time on a terminal. You can page forward with **[SPACE]**, backward with '**b**' and quit the manual page at any time with '**q**'.

1.5 Listing Files and Directories

The **ls** command will list all the files in the specified directories. You can use the **-l** option for a long listing format which displays information like size, permissions, owner, group, creation date, etc. If you do not specify a directory, by default it uses the current directory. The **-R** option causes **ls** to list recursively.

Some examples to try:

```
$ ls
$ ls -l
$ ls /
$ ls -l /
```

1.6 Editing Files

You can use various text editors either directly from the desktop or the terminal. Some editors are strictly text based and some have a graphical interface in X. Many editors are available such as: **emacs, vi, vim, gedit**, etc. Editors like **emacs** and **vi** / **vim** are not so easy to use but are fairly powerful and popular with

programmers. You are encouraged to learn **`vi`**/**`vim`** or **`emacs`**, many tutorials are available on the web. (Note: **`vim`** and **`emacs`** are available on Windows too).

If you are new to **`Unix`**, you can use **`gedit`** / **`nano`** which is more user friendly.

1.7 Logging Out

When you are done, you **MUST** log out from Linux. From the **`System`** menu, choose **`logout`** (if you really want to power down, you can choose **`shutdown`**, but that is not needed for normal lab session). It is not necessary to switch off the PC, but please switch off the monitor to conserve monitor life.

1.8 Backup and Transferring Files

You can transfer the files out of the lab PC by using the network, e.g. store on any cloud storage or use the **`scp`** (ssh file transfer) to transfer to **sunfire** server.

You can also use a USB flash drive to backup your work. When you insert a USB drive, it should be "*automatically mounted*", i.e. you should see the drive appearing on the "Launch Bar" on the left after a small delay. **Remember to unmount the USB drive before removing it**! This is analogous to the "Eject Drive" action in windows. From the desktop, just right-click the drive and select **`unmount.`** You can also mount and unmount manually from the command line interface, i.e. **`umount /dev/sdb1`**). **Without the unmount operation, your files may not be saved.**

> **Hard Disk Space Quota Note:**
>
> Since you are given only a **meagre 250Mb hard disk space** on the OS Lab account, please manage your space carefully. One of the main culprits is the use of web browser, which caches significant amount of info and can quickly overshoot your disk quota. Just clear the web browser cache whenever you run out of space.

1.9 Some Things to Remember!

The following points are to be noted for Unix in general and also Linux:

- Unix is *case sensitive*. Most commands are lowercase.
- Unlike Windows, Unix **has no drive letters** (i.e. no C:, D:, E: etc). Everything is in some directory.
- Unix uses forward slash (**`/`**) to separate directory names, while Windows uses backslash (**`\`**).

- The ***** wildcard is treated uniformly by Unix shells. In Windows, ***** works differently for different programs.
- It is worthwhile to look first at the man page of a command
- The default web browser is Firefox
- Try using various GUI and command line programs. Some programs need administrator privileges (in Unix, this is the root user who has *superuser privileges*) to run.

Section 2 Sample Usage Session

Skip this section if you already know how to use Unix / Linux.

1. **`mkdir junk`** // makes a new directory junk
2. **`cd junk`** //change directory to it, you are now in junk
3. **`pwd`** //prints the path of your current directory
4. **`echo abcd > test1.txt`** // makes a new file **`test1.txt`** with contents "**`abcd`**"
5. **`less test1.txt`** //display **`test1.txt`** using the pager. `**`q`**' will quit from less and `**`h`**' will give the help screen.

Select an editor to use (see previous editing section).

1. edit the file **`test1.txt`**, eg.
   `gedit test1.txt` //or any editor of your choice
2. change the content of the file "**`abcd`**" to something else
3. save the file in the editor (this is editor specific)
4. **`cat test1.txt`** //concatenate 1 or more files, and print to output, so this displays **`test1.txt`** because output is the terminal
5. **`cat test1.txt test1.txt > test2.txt`** //**`test1.txt`** is concatenated twice and the output saved to **`test2.txt`**
6. **`cp test1.txt test3.txt`** //copies **`test1.txt`** to **`test3.txt`**
7. **`ls`** //listing shows the 3 _les
8. **`rm test1.txt`** //deletes **`test1.txt`**
9. **`ls`** //only **`test2.txt`** and **`test3.txt`**
10. **`rmdir ../junk`** //try to remove directory, complains about non-empty directory
11. **`rm test*.txt; cd .. ; rmdir junk`** //no more _les in directory, go up to parent directory, remove junk directory. You can perform multiple commands in a single line by using the ";" separator.
12. **`logout`** //you can also use the GUI System Menu for this